Evaluation of Safe Control Policies for Mobile Robots in Dynamic Environments

Varun Murali, Niharika Arora, Ian Buckley

***Abstract*—With the growth of domestic robot industry, it is important to understand the navigation of mobile robots in dynamic environments. Use of mobile robots in the home can greatly improve the quality of life and is a growing field of research. With an increased interaction between humans and robots, and an increasingly shared workspace, it is important to study the safety of the robotic system and the extent of guarantees that can be made to minimize risk of injury. We present a nonlinear control approach to the navigation problem that achieves the objective of reaching pre-defined locations with provable guarantess of safety. Verification of the controller is performed first in MATLAB simulation, which shows that the controller is capable of driving the system to a pre-defined location while mainting all required safety rules defined in it's environment. The system is further investigated on the robot simulator, Gazebo and implemented on a Segway RMP 200 based robot, demonstrating that the nonlinear navigation controller successfully traverses the gap between theory and practice.**

I. Introduction

Safety is an important concern for mobile robots working in collaborative environments. Typical robotic systems seperately deal with safety and the task in a heirarchial fashion. A high level controller decides which control law to use based on the situation. Navigation is of fundamental concern for mobile robots. The ability to successfully avoid obstacles, both static and dynamic, while moving towards a goal location largely determines the utility of a mobile robotic platform. It is critical that robots avoid obstacles for their own safety as well as the safety of the people around them. We use the navigation problem to motivate the addition of safety directly into the controller for the robot.

The navigation control protocol will first be motivated and derived. After explaining the protocol, MATLAB simulation will be used to verify that the navigation control protocol correctly drives the state of the robot to a target. After demonstrating that the robot can navigate to a target location, a trajectory will be generated for a map using a probabilistic roadmap, and the control law will be used to drive the robot along the trajectory. After thoroughly verifying the navigation control protocol in simulation, the controller will be implemented in the ROS framework and evaluated on the mobile robot Jeeves which will verify the navigation control protocol in a real world scenario.

II. Approach

This section details the development of the safe navigation control protocol. In particular, the soft and hard constraints previously presented motivate the use of control Lyapunov functions and barrier functions for achieving safe navigation.

A. Soft Constraints as Control Lyapunov Functions

To design the control law, the standard quadratic Lyapunov function candidate is first considered:

$$V = \frac{1}{2}(r^2 + \theta^2) \tag{1}$$

$$\dot{V} = r\dot{r} + \theta\dot{\theta} \tag{2}$$

$$= -rv\ \cos(\delta) + \theta v\ \sin(\delta) \tag{3}$$

Fixing δ to enforce a virtual steering control as in [3],

$$\delta = \arctan(-k_1\theta). \tag{4}$$

The derivative of the Lyapunov function becomes:

$$\dot{V} = -rv\ \cos(\arctan(-k_1\theta)) + \theta v\ \sin(\arctan(-k_1\theta)). \tag{5}$$

Because $\theta \in (-\pi, \pi]$, $v > 0$, and $r \geq 0$, $\dot{V} < 0\ \forall \theta, r \neq 0$; thus, the steering control asymptotically drives $r, \theta \to 0$. By choosing $v = k_3 r$ in some neighborhood of $r = 0$ for positive constant k_3, the singularity at $r = 0$ is removed and the system is globally asymptotically stable. Furthermore, by the definition of the steering control, $\delta \to 0$ as $\theta \to 0$, so the virtual steering control not only drives the robot to the final position, but aligns the heading of the robot with the positive x-axis of the global coordinate frame by the time it arrives at the final position.

To drive the state to zero through the steering control, the heading must be driven such that $\delta = \arctan(-k_1\theta)$. An objective z_1 is defined:

$$z_1 \equiv \delta - \arctan(-k_1\theta). \tag{6}$$

The derivative is calculated to yield:

$$\dot{z_1} = \left(1 + \frac{k_1}{1 + (k_1\theta)^2}\right)\frac{v}{r}\ \sin(z_1 +\ \arctan(-k_1\theta)) + \omega. \tag{7}$$

A specific example of a control law that can achieve this objective is shown through feedback linearization by choosing the angular velocity

$$\omega = -\frac{v}{r}\left[k_2 z_1 + \left(1 + \frac{k_1}{1 + (k_1\theta)^2}\right)\sin(z_1 +\ \arctan(-k_1\theta))\right], \tag{8}$$

the objective dynamics are globally exponentially stable with the following form:

$$\dot{z_1} = -k_2\frac{v}{r}z_1, \tag{9}$$

where k_1 and k_2 can be chosen to increase or decrease how aggressive the steering control is. This is modelled along with other constraints by defining:

$$V_2 = \frac{1}{2}z^2 \tag{10}$$

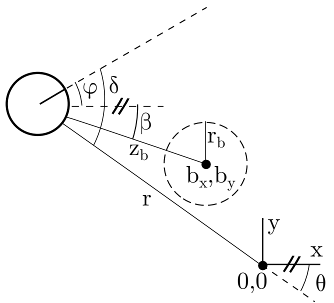


Fig. 1: The figure above shows the egocentric co-ordinate system of the robot. In addition, the obstacle is visulaized. The angle β is the angle the vector joining the center of the robot to the obstacle makes with respect to the x-axis of the global reference frame.

A second steering controller is designed to drive the state away from the origin and is defined by the following objective:

$$z_2 = \delta - \text{ atan}(k_1\beta), \tag{11}$$

where β is the angle formed between the global positive x-axis and the vector z_b pointing from the robot to the obstacle. Figure 1 shows the egocentric coordinate system with the addition of the obstacle, where b_x, b_y is the global coordinate of the obstacle and r_b is the minimum safe distance.

The derivative of z_2 is similarly computed as with z_1, and is given by

$$\dot{z_2} = \dot{\delta} \tag{12}$$

By choosing the angular velocity input w to drive the heading $\delta = \text{ arctan}(-k_1\theta)$, steering control, which drives the state to zero, is achieved. Similarly, choosing the angular velocity input w to drive the heading $\delta = \text{ arctan}(k_1\beta)$ causes the robot to drive away from zero. Thus, there exists a control input that satisfies the control Lyapunov function $\dot{V} < 0$; furthermore, following Definition 3 in [1], the control Lyapunov function is exponentially stabilizing if it satisfies

$$\inf_{u\in U}\left[\dot{V} + \gamma V\right] \leq 0 \tag{13}$$

for some constant $\gamma > 0$, so choosing the correct inputs u will achieve the desired exponential stabilization. According to [2], by picking a soft constraint relaxation variable λ, the condition for exponential stabilization shown in (13) is relaxed and can be enforced simultaneously along with a single hard constraint. Because control of the robotic system requires either $\dot{z_1} = -k_2\frac{v}{r}z_1$ or $\dot{z_2} = -k_2\frac{v}{r}z_2$ and $\dot{V} < 0$, two soft constrained expressions must be satisfied at a time to achieve exponential stabilization of the state to zero.

B. Barrier Functions

As in [2] and [1], control barrier functions are a logical way of enforcing safety constraints in navigation from a nonlinear control perspective.

A zeroing barrier function is used to drive the robot away from obstacles if it leaves the safe set (e.g. the robot gets within an unsafe threshold distance of the obstacle). The safe set C is described by the following expressions:

$$C = x \in \mathbb{R}^3 | z_b \geq z_{safe} \tag{14}$$

$$\partial C = x \in \mathbb{R}^3 | z_b = z_{safe} \tag{15}$$

$$\text{Int}(C) = x \in \mathbb{R}^3 | z_b > z_{safe} \tag{16}$$

As presented in [4], a barrier function is a zeroing barrier function $h(x)$ if $\dot{h}(x) \leq \gamma h(x)$. For the purpose of avoiding obstacles in the event that the robot leaves the safe set, a zeroing barrier function was chosen with the following form:

$$h(x) = \sqrt{(-v\,\cos(\theta) - b_x)^2 + (v\,\sin(\theta) - b_y)^2}. \tag{17}$$

The derivative of $h(x)$ is given by the following expression:

$$\begin{aligned}\dot{h}(x) = \frac{1}{h(x)}[(-v\,\cos(\delta)(r - b_y\,\sin(\theta) + b_x\,\cos(\theta)) \\ -v\,\sin(\delta)(b_x\,\sin(\theta) + b_y\,\cos(\theta))]\end{aligned} \tag{18}$$

A barrier function was chosen to prevent the robot from leaving the safe set. According to Definition 2 in [2], the barrier function B is a control barrier function if

$$\inf_{u\in U}\left[\dot{B} + \frac{\gamma}{B}\right] \leq 0. \tag{19}$$

Thus, by satisfying the equation $\dot{B} \leq 1/B$, the barrier function will prevent the robot from leaving the safe set. The barrier function was chosen to be

$$B = \frac{1}{h(x) - z_{safe}}, \tag{20}$$

and its derivative is given by

$$\dot{B} = \frac{-\dot{h}(x)}{(h(x) - z_{safe})^2}. \tag{21}$$

C. QP Based Controller

Quadratic programming is used to find control inputs that simultaneously satisfy both the soft and hard constraints of the system. To minimize control inputs, the cost function $\mathbf{u}^T H\mathbf{u} + F^T\mathbf{u}$ is chosen. The QP is given by the following expression:

$$u^*(x,z) = \underset{\mathbf{u}=\begin{bmatrix} v \\ \omega \\ \lambda_1 \\ \lambda_2 \\ \alpha_1 \\ \alpha_2 \end{bmatrix}\in\mathbb{R}^6}{\arg\min}\ \frac{1}{2}\mathbf{u}^T H\mathbf{u} + F^T\mathbf{u} \tag{22}$$

Parameters	Value
c_a	0.0
c_b	2.0
c_c	-0.1
c_d	0.1
γ	1

TABLE I: The table above shows the values that we used for matlab, gazebo and real experiments

$$\text{s.t.} \quad \dot{V_1} - \alpha_1 V_1 \leq \lambda_1 \tag{23}$$
$$\dot{V_2} - \alpha_2 V_2 \leq \lambda_2 \tag{24}$$
$$h(\dot{x}) - \alpha h(x) \leq \lambda_1 \tag{25}$$
$$\dot{B} \leq \gamma / B \tag{26}$$
$$c_a \leq v \leq c_b \tag{27}$$
$$c_c \leq \omega \leq c_d \tag{28}$$
$$\alpha_1, \alpha_2 \leq 0 \tag{29}$$

In (22), the matrix $H \in \mathbb{R}^{6\times 6}$ has the form

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{30}$$

and the vector $F^T \in \mathbb{R}^{1\times 6}$ has the form

$$F^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 10 & 1 \end{bmatrix}. \tag{31}$$

The last two elements in the diagonal of H were chosen to be zero to ensure that the negative quadratic cost of the slack variables α_1 and α_2 is as large as possible. Conversely, the last two elements of F^T were chosen to ensure that α_1 and α_2 were negative to minimize the cost.

The control Lyapunov functions V_1 and V_2 are the Lyapunov functions corresponding to the objectives of driving the state to zero and enforcing the steering control respectively; the equations are given by

$$V_1 = \frac{1}{2}(r^2 + \theta^2) \tag{32}$$

and

$$V_2 = \frac{1}{2} z_*^2. \tag{33}$$

z_* in V_2 represents the different steering objectives z_1 and z_2 in (6) and (11). Because there are two steering objectives, two QP control laws are required: one which drives the state towards the target while avoiding obstacles, and the other that drives the state in such a way that the robot steers away from obstacles. Toggling between steering objectives is dependent on the obstacle measurement. If the measurement of the distance between the robot and the obstacle was less than the minimum safe distance, then the steering objective z_2 is used. Otherwise, z_1 is used, and $\alpha_1 = 0$ is additionally enforced. $h(x)$ and B in the QP controller are given in equations (18) and (20).

III. Experimental Results

Real robot experiments were performed on the segway robot "jeeves". The controller was implementd in the ROS framework. The hokuyo laser scans are subscribed to and the appropriate processing is performed. The robot is commanded to drive to a set point and the optimal u is computed at every time step. The dynamics is propogated forward using the odometry. To add the obstacle, a human moved into the operation range of the obstacleas can be seen in Figure 3 and ensured that the robot did not break any safety constraints. A video showing one experimental run can be seen at https://youtu.be/uhTiYyZxtfc.

The video shows that the robot maintains its safe distance from the human and tries to avoid the human, when the human walks out of the radius of influence the robot is free to move at greater speeds again.

IV. Conclusion

We show that the nonlinear controller is generatlizable to multiple robots and is provably safe. The theoretical analysis shows that the controller stabilizes the system while obeying the "safety" rules. The exaperimental analysis shows that the system can be implemented on a real world system and the results are consistent with the claims. The developed system is almost self-contained and does not require many parameter adjustments. For future work, we propose more complex modelling of the obstacles to better capture the nature of the obstacle. For example, the safety barrier around a human is different from that around a wall. It is more "socially" acceptable to drive close to a wall but not as acceptable to be that close to a human.

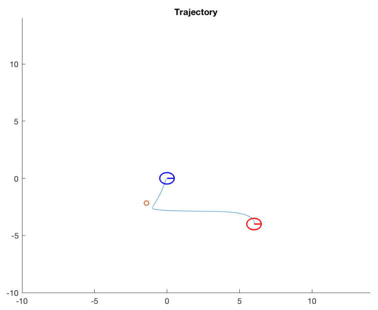
References

[1] Aaron D. Ames, Kevin Galloway, Koushil Sreenath, and Jessy W. Grizzle. Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. In *IEEE Transactions on Automatic Control (Volume:59 , Issue: 4)*, pages 876–891. IEEE, 2014.

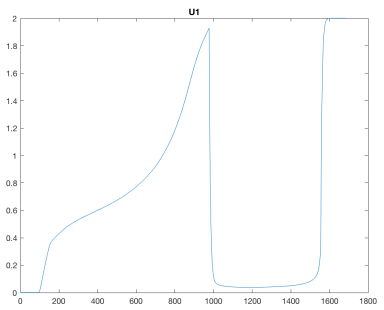
[2] Aaron D. Ames, Jessy W. Grizzles, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control*, pages 6271–6278. IEEE, 2014.

[3] Jong Jin Park and Benjamin Kuipers. A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4896–4902. IEEE, 2011.

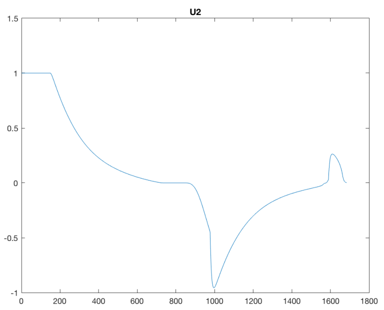
[4] Xiangru Xu, Paulo Tabuada, Jessy W. Grizzle, and Aaron D. Ames. Robustness of control barrier functions for safety critical control. In *Analysis and Design of Hybrid Systems(Volume 48)*, pages 54–61, 2015.



(a) Trajectory of the robot



(b) The linear velocity of the robot



(c) The angular velocity of the robot

Fig. 2: The above figures show the trajectory, linear velocity and angular velocity of the robot while avoiding the obstacle.

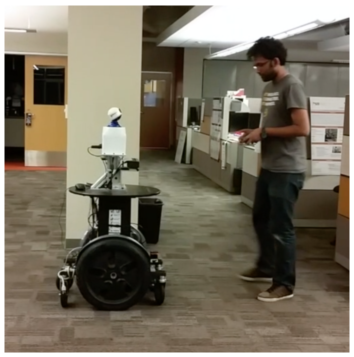


Fig. 3: Real robot experiment with the segway RMP robot. The robot is stopped at a safe distance from the human obstacle.